



# **EOSDIS**

NASA'S EARTH OBSERVING SYSTEM  
DATA AND INFORMATION SYSTEM

# **HDF Update**

Elena Pourmal

The HDF Group

[epourmal@hdfgroup.org](mailto:epourmal@hdfgroup.org)

This work was supported by NASA/GSFC under  
Raytheon Co. contract number NNG15HZ39C

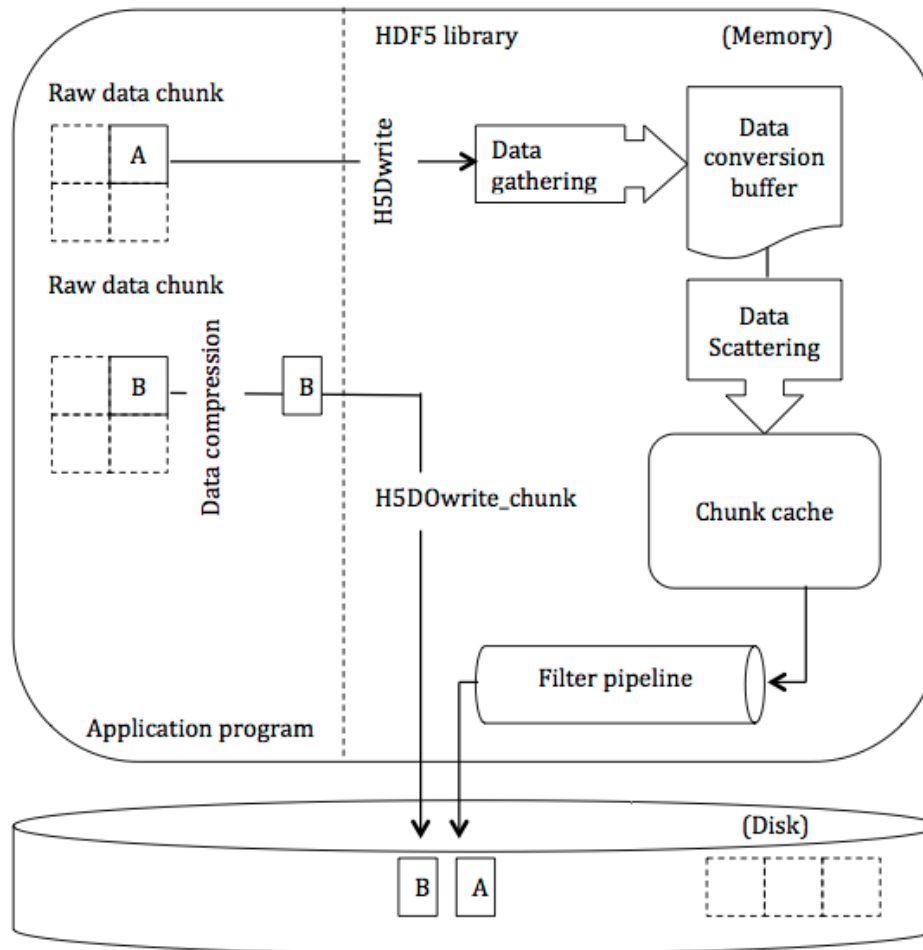
# Outline

- What's is new in HDF?
- HDF tools
  - HDFView
  - Nagg
  - ODBC
- Q and A: Tell us about your needs

# HDF5

- HDF5 Compression
  - Faster way to write compressed data to HDF5
  - Community supported compression filters
- Single writer/multiple reader file access
- Virtual Data Set
- HDF5 JNI is part of the HDF5 source code

# Direct chunk write: H5DOWrite\_chunk



# Performance results for H5DOWrite\_chunk

Test result on Linux 2.6, x86\_64  
Each dataset contained 100 chunks,  
written by chunks

Dataset size (MB)	95.37		762.94		2288.82	
Size after compression (MB)	64.14		512.94		1538.81	
Dataset dimensionality	100x1000x250		100x2000x1000		100x2000x3000	
Chunk dimensionality	1000x250		2000x1000		2000x3000	
Datatype	4-byte integer		4-byte integer		4-byte integer	
	speed <sup>1</sup>	time <sup>2</sup>	speed	time	speed	time
H5Dwrite writes without compression filter	77.27	1.23	97.02	7.86	91.77	24.94
H5DOWrite_chunk writes uncompressed data	79	1.21	95.71	7.97	89.17	25.67
<b>H5Dwrite writes with compression filter</b>	<b>2.68</b>	<b>35.59</b>	<b>2.67</b>	<b>285.75</b>	<b>2.67</b>	<b>857.24</b>
<b>H5DOWrite_chunk writes compressed data</b>	<b>77.19</b>	<b>0.83</b>	<b>78.56</b>	<b>6.53</b>	<b>96.28</b>	<b>15.98</b>
Unix writes compressed data to Unix file	76.49	0.84	95	5.4	98.59	15.61

1 Speed in MB/s

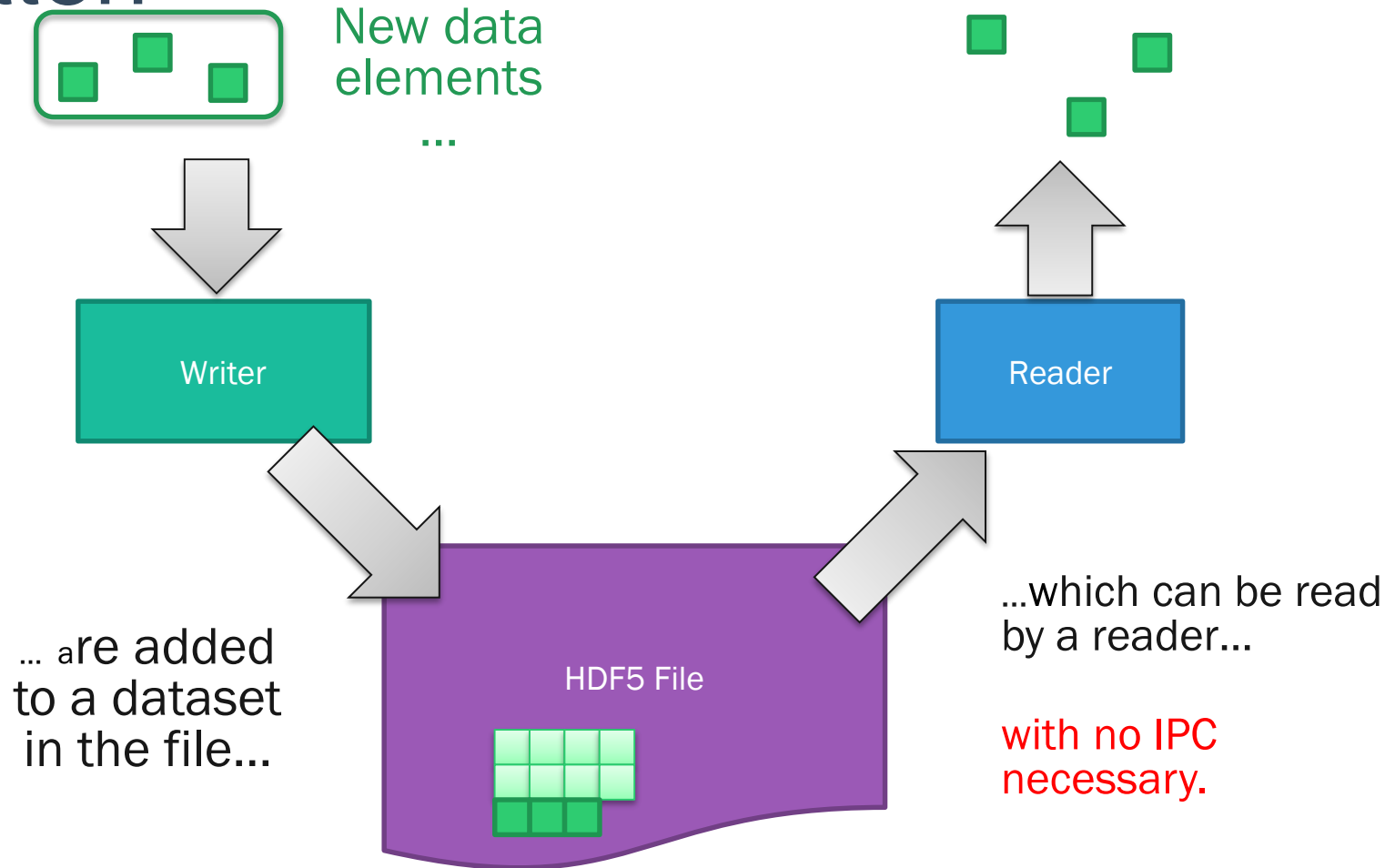
2 Time in seconds

# Dynamically loaded filters

- Problems with using custom filters
  - “Off the shelf” tools do not work with the third-party filters
- Solution
  - Use a 1.8.11 and later and dynamically loaded HDF5 compression filters
  - Maintained library of HDF5 compression filters
- <https://github.com/nexusformat/HDF5-External-Filter-Plugins>

# Example: Choose compression that works for your data

# SWMR: Data access to file being written





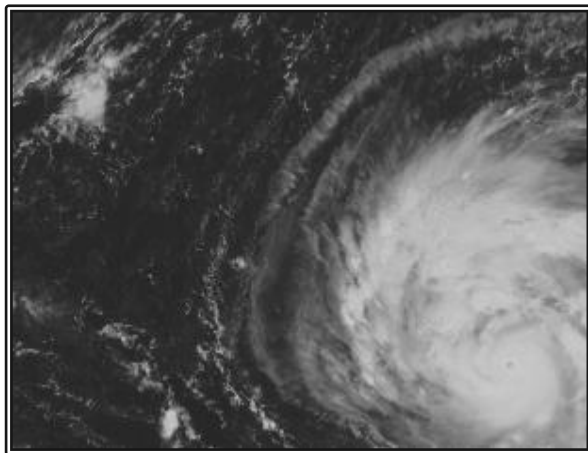
# SWMR

- Released in HDF5 1.10.0
- Restricted to append-data only scenario
- SWMR doesn't work on NFS
- Files are not compatible with HDF5 1.8.\* libraries
- Use h5format\_convert tool
  - Converts HDF5 metadata in place
  - No raw data is rewritten

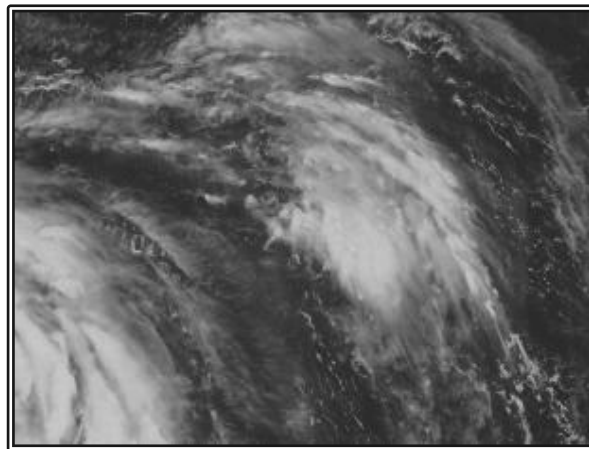
# VDS

- Data stored in multiple files and datasets can be accessed via one dataset (VDS) using standard HDF5 read/write

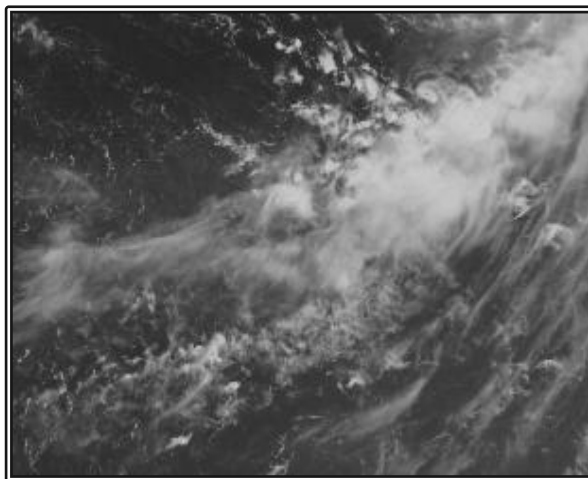
# Collect data one way ....



File: a.h5  
Dataset /A



File: b.h5  
Dataset /B



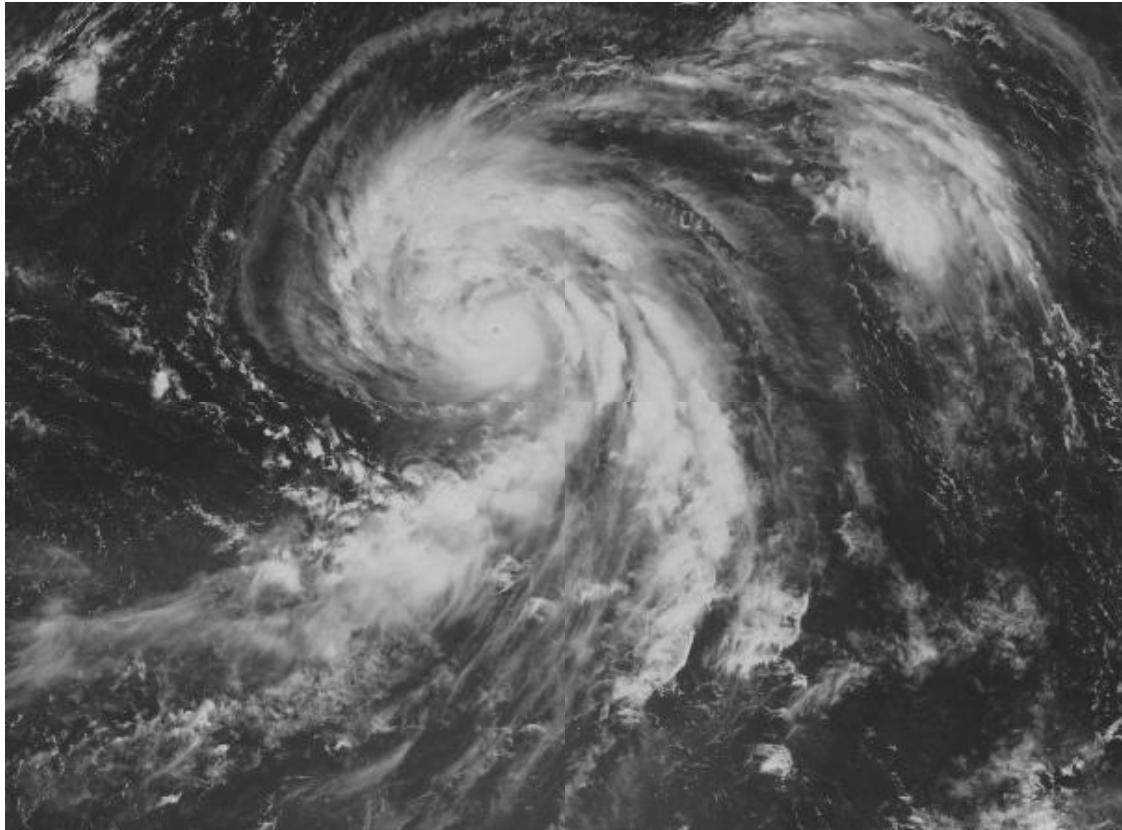
File: c.h5  
Dataset /C



File: d.h5  
Dataset /D

# Present it in a different way...

Whole image



File: F.h5  
Dataset /D

# VDS

- VDS works with SWMR
- File with VDS cannot be accessed by HDF5 1.8.\* libraries
- Use h5repack tool to rewrite data (1.10.0-patch1)

# HDF5 Roadmap for 2016 -2017

- May 31 -HDF5 1.10.0-patch1
  - h5repack, Windows builds, Fortran issues on HPC systems
- Mid-summer HDF5 1.10.1
  - Address issues found in 1.10.0
- December
  - HPC features that didn't make it into 1.10.0 release
- Maintenance releases of HDF5 1.8 and 1.10 versions (May and November)

# HDF4

- HDF 4.2.12 (July 2016)
- Support for latest Intel, PGI and GNU compilers
- HDF4 JNI included with the HDF4 source code

# HDF5 Roadmap for 2016 -2017

- December 2016
  - Minor bug fixes (if required)
- Summer 2017
  - Keep up with computing environment



# HDFView

- HDFView 2.13
  - Bug fixes
  - Based on HDF5 1.8 releases
- HDFView 3.0-alpha
  - New GUI
  - Better internal architecture
  - Based on HDF5 1.10 release

# HDFView 3.0 Screenshot

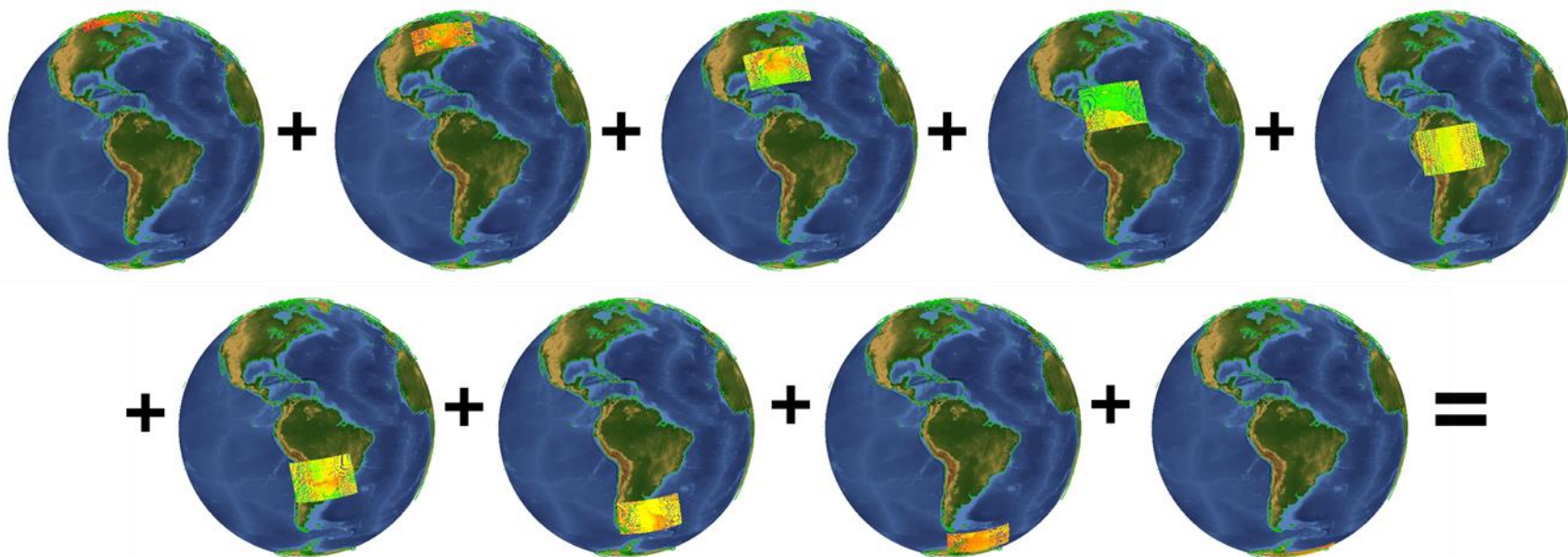
- Add to demonstrate some new feature (compound DT?)

# Nagg tool

Nagg is a tool for rearranging NPP data granules from existing files to create new files with a different aggregation number or a different packaging arrangement.

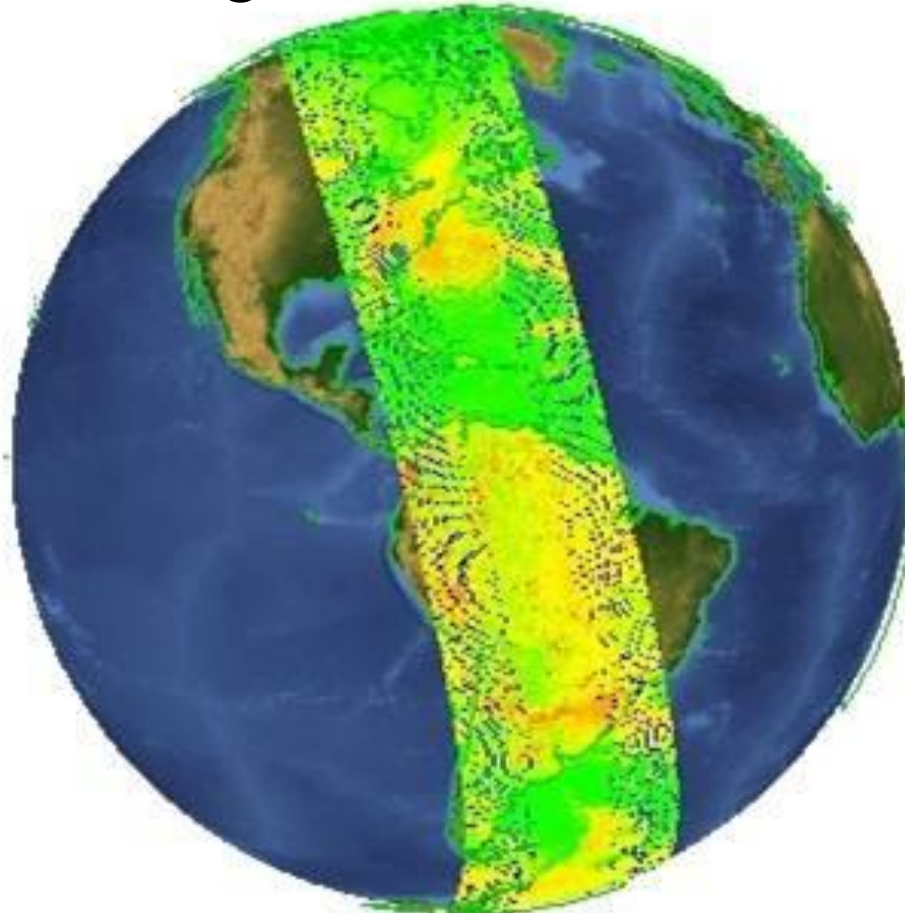
# Nagging Illustration - IDV visualization

rules each in GIMDO

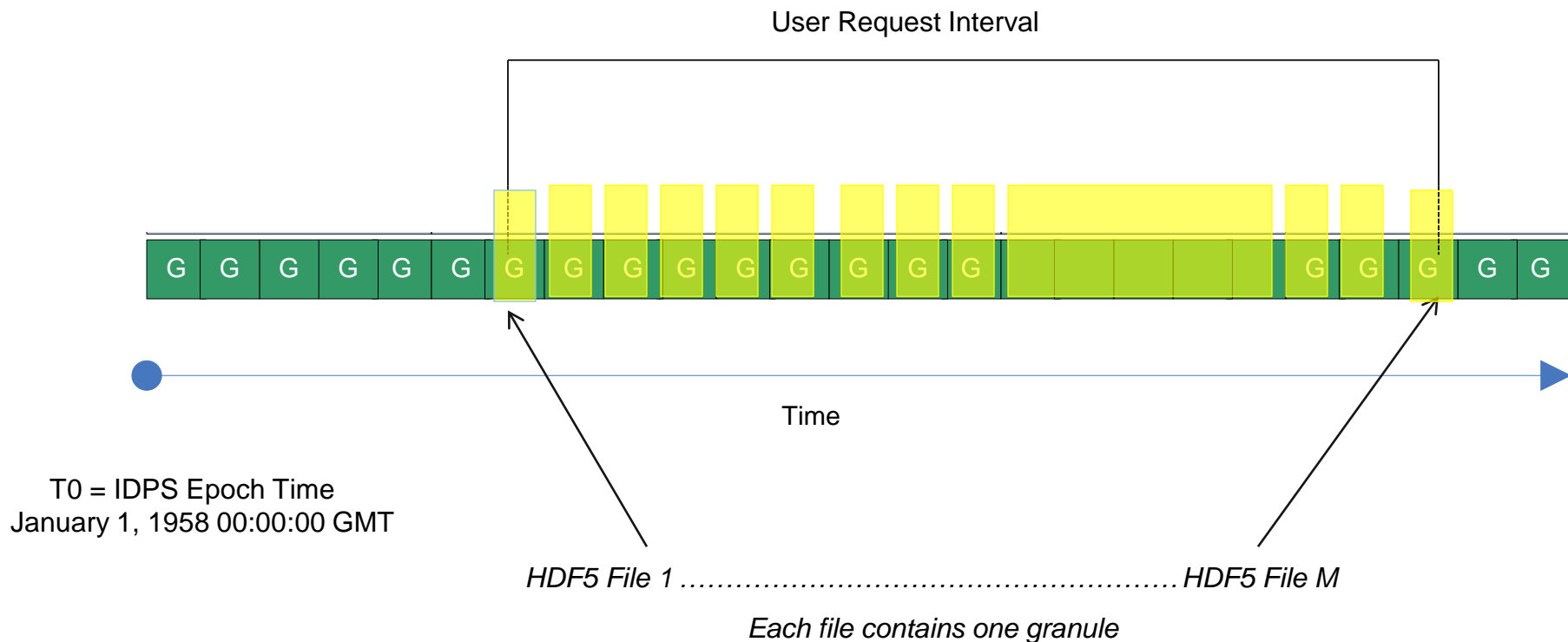


# Nagg Illustration - IDV visualization

1 output file –36 granules in GMODO-SVM07... file



# nagg: Aggregation Example

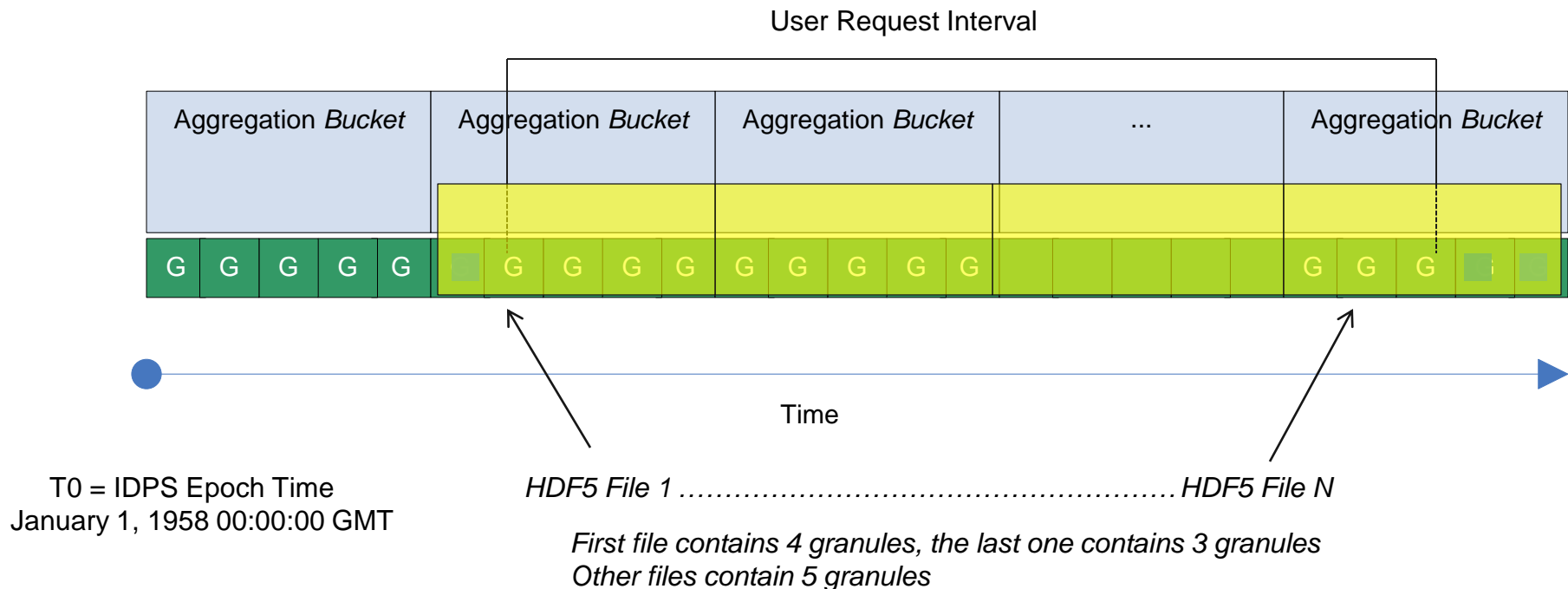


- User requests data from the IDPS system for a specific time interval
- Granules and products are packaged in the HDF5 files according to the request
- This example shows one granule per file for one product

# nagg: Aggregation Example

Example: `nagg -n 5 -t SATMS SATMS_npp_d2012040*.h5`

Nagg copies data to the newly generated file(s).

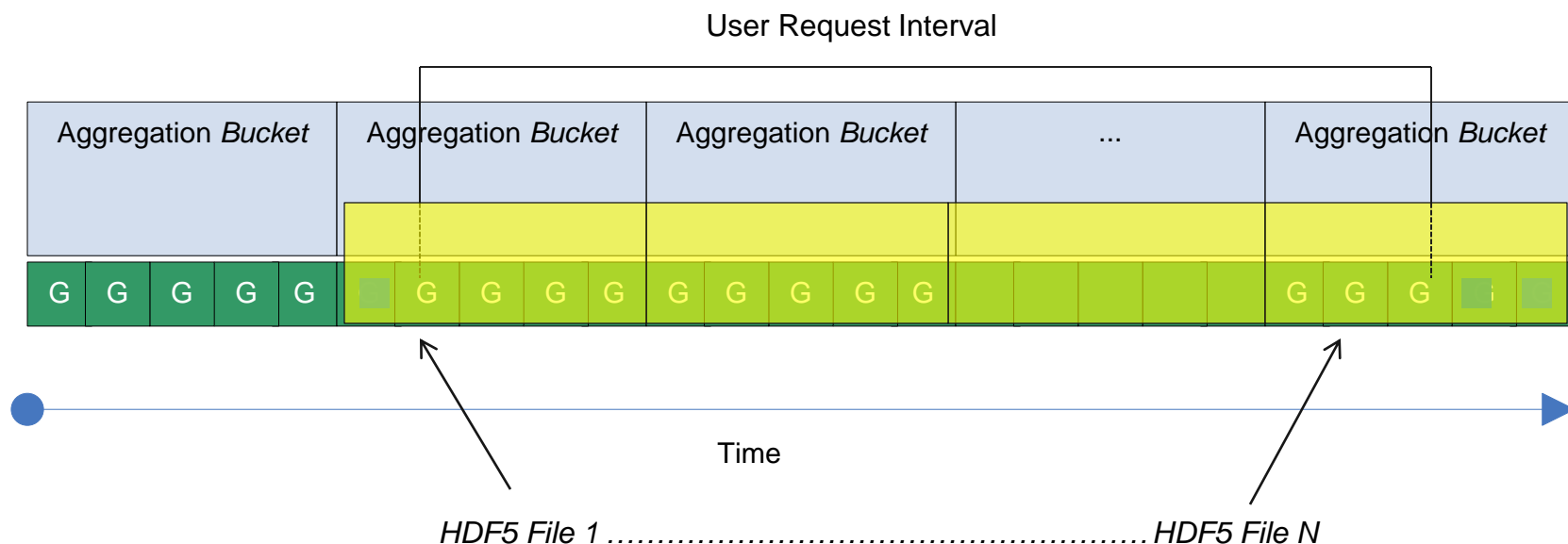


- Produced files co-align with the aggregation bucket start
- HDF5 files are 'full' aggregations (full, relative to the aggregation period)
- Geolocation granules are aggregated and packaged; see `-g` option for more control

# Possible enhancement

Example: `nagg -n 5 -v -t SATMS SATMS_npp_d2012040*.h5`

Nagg with `-v` option doesn't copy data to the newly generated file(s).

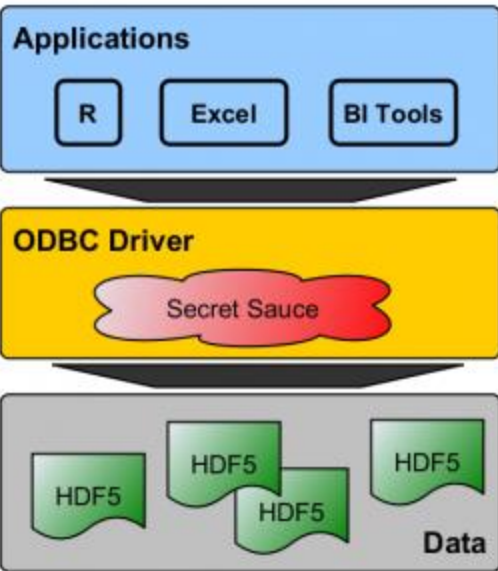


*Each file contains a virtual dataset. First file contains a dataset **mapped** to 4 granules, the last one contains a virtual dataset **mapped** to 3 granules  
Other files contain virtual datasets; each dataset is **mapped** to 5 granules*

- NO RAW DATA IS REWRITTEN
- Space savings
- No I/O performed on raw data



# HDF5 ODBC Driver

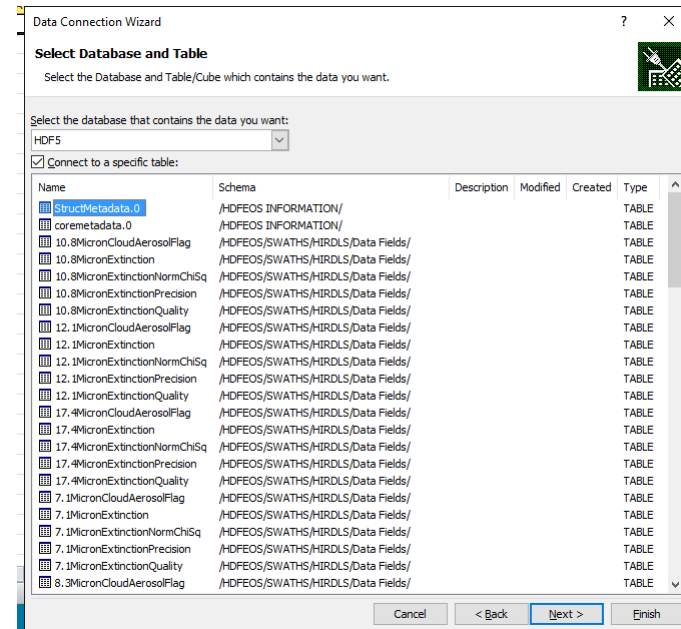
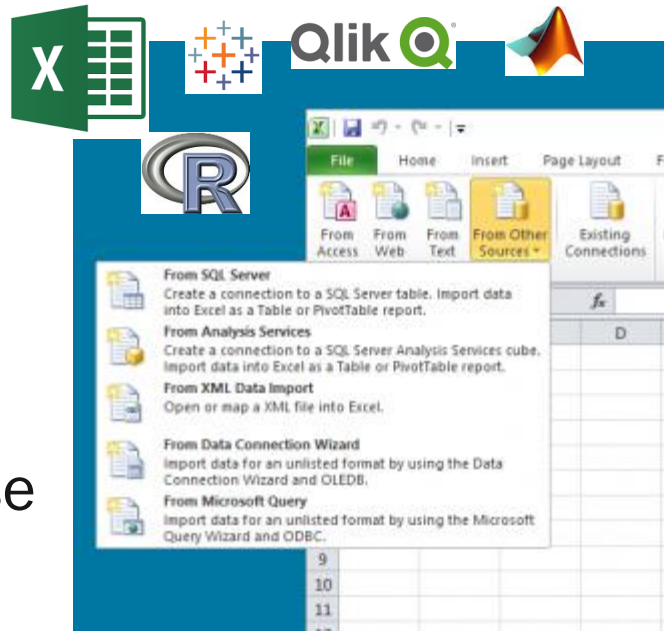


- ✓ Tap into the **USB bus of data** (ODBC)
- ✓ Direct access to your HDF5 data from your favorite BI application(s)

[odbc@hdfgroup.org](mailto:odbc@hdfgroup.org)

- Join the Beta
- Tell your friends
- Send feedback

- Beta test now
- Q3 2016 Release
- Desktop version
- Certified-for-Tableau
- Client/server version this Fall



# New requirements and features?

- Tell us your needs (here are some ideas):
  - Multi-threaded compression filters
  - H5DOread\_chunk function
  - Full SWMR implementation
  - Performance
  - Backward/forward compatibility
- Other requests?

This work was supported by  
NASA/GSFC under Raytheon Co.  
contract number NNG15HZ39C

**Raytheon**